MASTER COPY KEEP FOR REPRODUCTION PURPOSES

# REPORT DOCUMENTATION PAGE

Form Approved
OMB NO. 0704-0188



| 1. AGENCY USE ONLY *(Leave blank)* | 2. REPORT DATE | 3. REPORT TYPE AND DATES COVERED |
|---|---|---|
| | October 1997 | Technical Report |

**4. TITLE AND SUBTITLE**
Optimization of Placement Driven by the Cost of Wire Crossing

**5. FUNDING NUMBERS**
DAAH04-94-G-0280

**6. AUTHOR(S)**
N. Kapur, D. Ghosh, F. Brglez

**7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(ES)**
Collaborative Benchmarking Laboratory
Department of Computer Science
North Carolina State University
Box 7550 NCSU
Raleigh, NC 27695-7550

**8. PERFORMING ORGANIZATION REPORT NUMBER**

**9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)**
U.S. Army Research Office
P.O. Box 12211
Research Triangle Park,, NC 27709-2211

**10. SPONSORING / MONITORING AGENCY REPORT NUMBER**
ARO 33616.7-EL

**11. SUPPLEMENTARY NOTES**
The views, opinions and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy or decision, unless so designated by other documentation.

**12a. DISTRIBUTION / AVAILABILITY STATEMENT**
Approved for public release; distribution unlimited.



**13. ABSTRACT** *(Maximum 200 words)*

We conjecture that good column-based placements can be produced by minimizing two wire crossing numbers:

(1) the total wire crossing of all edges between cells in the wiring channel, and
(2) the maximum wire crossing on the imaginary cutlines that separate cells on the opposite ends of the channels.

We leverage the canonical form of the multi-level bipartite directed graph to formalize a unit-grid model that allows us to define and evalutate parameters such as total wire crossing, critical wire crossing, total wire length, critical wire length, critical wire density, total wire density, as well as height, width, and area of the embedded graph.

We implemented a prototype placement algorithm TOCO that minimizes the cost of wire crossing, and a universal unit-grid based placement evaluator place_eval. We have designed a number of statistical experiments to demonstrate the feasibility and the promise of the proposed approach.

| 14. SUBJECT TERMS | 15. NUMBER IF PAGES |
|---|---|
| wire crossing, wire length, cell placement, bipartite graph model, design of experiments, benchmarking | 10 |
| | 16. PRICE CODE |

| 17. SECURITY CLASSIFICATION OR REPORT | 18. SECURITY CLASSIFICATION OF THIS PAGE | 19. SECURITY CLASSIFICATION OF ABSTRACT | 20. LIMITATION OF ABSTRACT |
|---|---|---|---|
| UNCLASSIFIED | UNCLASSIFIED | UNCLASSIFIED | UL |

CBL (Collaborative Benchmarking Laboratory)
Department of Computer Science
Campus Box 7550
North Carolina State University
Raleigh, NC 27695

# Optimization of Placements Driven By the Cost of Wire Crossing

Nevin Kapur Debabrata Ghosh Franc Brglez

Technical Report 1997-TR@CBL-08-Kapur
October 1997


If you choose to cite this report, please add the following entry to your bibliography database:

```
@techreport{
1997-TR@CBL-08-Kapur,
author = "N. Kapur and D.Ghosh and F. Brglez",
title  = "{Optimization of Placements Driven By the Cost of Wire Crossing}",
institution = "{CBL, CS Dept., NCSU, Box 7550, Raleigh, NC 27695}",
number = "1997-TR@CBL-08-Kapur",
month  = "Oct",
year   = "1997",
note   = "{Also available at http://www.cbl.ncsu.edu/publications}"
}
```

To contact the Collaborative Benchmarking Laboratory via Internet, you may consider:

| | |
|---|---|
| WWW : | `http://www.cbl.ncsu.edu/` |
| Anonymous FTP : | `ftp://ftp.cbl.ncsu.edu` |
| For an auto-reply: | `benchmarks@cbl.ncsu.edu` |
| To deposit a file at CBL: | `ftp cbl.ncsu.edu`<br>`cd /pub/Incoming`<br>`put new_benchmark.tar.Z` |

Technical Report 1997-TR@CBL-08-Kapur (December 1997)

# Optimization of Placements Driven By the Cost of Wire Crossing

Nevin Kapur Debabrata Ghosh Franc Brglez
CBL (Collaborative Benchmarking Lab), Dept. of Comp. Science, Box 7550, NC State U., Raleigh, NC 27695
http://www.cbl.ncsu.edu/

**Abstract** – *We conjecture that good column-based placements can be produced by minimizing* two *wire crossing numbers: (1) the total wire crossing of all edges between cells in the wiring channel, and (2) the maximum wire crossing on the imaginary cutlines that separate cells on the opposite ends of the channels. We leverage the canonical form of the multi-level bipartite directed graph to formalize a unit-grid model that allows us to define and evalutate parameters such as total wire crossing, critical wire crossing, total wire length, critical wire length, critical wire density, total wire density, as well as height, width, and area of the embedded graph.*

*We implemented a prototype placement algorithm* `TOCO` *that minimizes the cost of wire crossing, and a universal unit-grid based placement evaluator* `place_eval`. *We have designed a number of statistical experiments to demonstrate the feasibility and the promise of the proposed approach.*

**Keywords:** wire crossing, wire length, cell placement, bipartite graph model, design of experiments, benchmarking.

## I. Introduction

Area and performance in submicron-technology VLSI circuits are critically dominated by the interconnect. The variations in interconnect that arise *after* cell placement and routing, such as wire crossings, wire density, wire loading, wire length, etc., may be hard if not impossible to predict. Similarly, the interconnect models associated with the logic implementation and logic testability *prior* to placement and routing may change considerably *after* cell placement and routing.

In row–based cell placement, the cost function that continues to dominate the research relates mostly to total wirelength, e.g. the simulated annealing approach [1]. The length of the interconnect in randomized and optimized placements has been formalized in [2]. Abstract placement models, such as presented in [3], also aim to minimize the channel density.

The crossing number and wire area have been investigated for effective wire lower bounds and edge length in a variety of computational VLSI circuits, using a *graph-based* grid model [4]. In this paper, we propose a new unit-grid model, derived from the a multi-level bipartite graph in canonical form [5]. We conjecture that good placements can be produced by minimizing *two* wire crossing numbers in our bipartite graph model: (1) the total wire crossing of all edges, and (2) the maximum wire crossing of all cell channel separator segments.

Crossing theory has been developed to improve the readability of hierarchical structures [6]. The problem for placing the nodes for minimum wire crossing is NP-complete [7], even for 2-layer graphs. Several heuristics – barycenter [8], median [9], assignment [10] – have been proposed to minimize the wire crossings. A survey of exact and heuristic algorithms is available in [11]. An improvement on the median heuristic has been described in [12] and we use the associated drawing program `DOT` to support our initial experiments reported in this submission. We are also investigating alternative approaches to minimize wire crossing, subject to constraints specific to VLSI circuits and expect to report on them in the final submission.

We argue that for layout problems in particular, case-by-case evaluations, however detailed, of a few unrelated benchmark circuits are not likely to reveal a set of statistically consistent results that can drive and support important improvements for the new generation of algorithms and tools. To illustrate our point, we analyze grid-based layouts as well as placed-and-routed layouts for a number of equivalence mutant classes introduced in [5][1]. For convenience of the reviewers, we copy the section on the design of experiments and the mutant classification we used in [5]. In the the final submission, this section will be edited to minimum size.

The paper is organized into the following sections :

(2) background and motivation;
(3) correlations in 2–layer graphs;
(4) notation and definitions;
(5) cell and net folding and compaction;
(6) design of experiments;
(7) summary of experiments;
(8) conclusions.

## II. Background and Motivation

We use a simple 11-node *planar circuit* to informally introduce the proposed unit-grid model and illustrate the conjecture that good placements can be produced by minimizing *two* wire crossing numbers in its directed bipartite graph model representation. A summary of our experiments is shown in Figure 1:

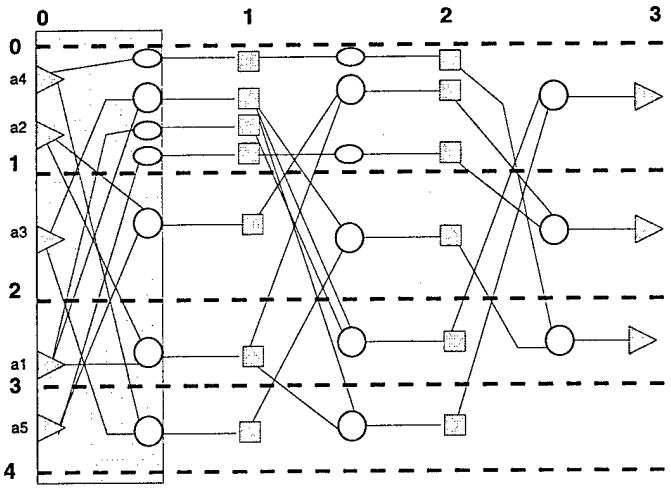
**(a)** The circuit is *not* drawn in the planar form. Rather, we draw it as an instance of a *random placement*, in a *canonical form*, of a bipartite directed acyclic graph [5]. The 'square' and 'triangle' nodes represent *net nodes*, the circles and ellipses are *cell nodes*. Each circle represents a *logic cell* in the original circuit, each ellipse represents a *feedthrough cell* introduced by the canonical form. We call the horizontal lines, drawn as dotted lines and indexed from 0 to 4, *cell separators*. We call the vertical lines that could be drawn through the net nodes and indexed from 0 to 3, *net separators*. The total wire crossing number for all edges in the graph as drawn here, is 54. Starting with the placement shown, not even the state-of-the-art algorithm such as `DOT` [12] can render the circuit planar (wire crossing is minimized to a value of 7).

**(b)** Analysis of the left-most net-to-cell channel in (a) shows a total wire crossing of 23, wire length of 23.5, and the critical wire density of 7. Wire crossing refers to

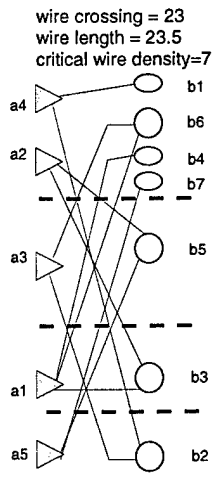
This research has been supported by contracts from the Semiconductor Research Corporation (94-DJ-553), SEMATECH (94-DJ-800), and DARPA/ARO (P-3316-EL/DAAH04-94-G-2080).



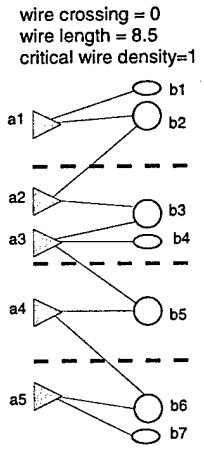
[1] Available as a preprint from http://www.cbl.ncsu.edu/publications/).



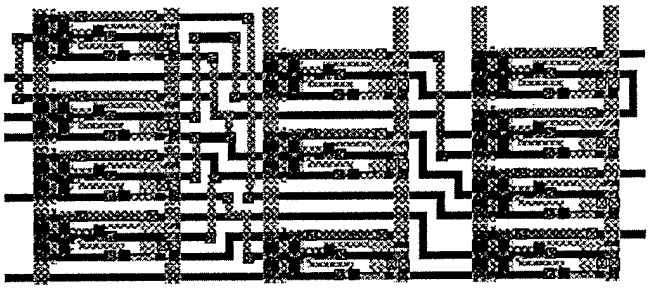

(a) Initial placement of circuit `planar' (mutant #2):
(initial wire crossing=54, optimized wire crossing=7)




(b) Example of a net-to-cell channel initial placement



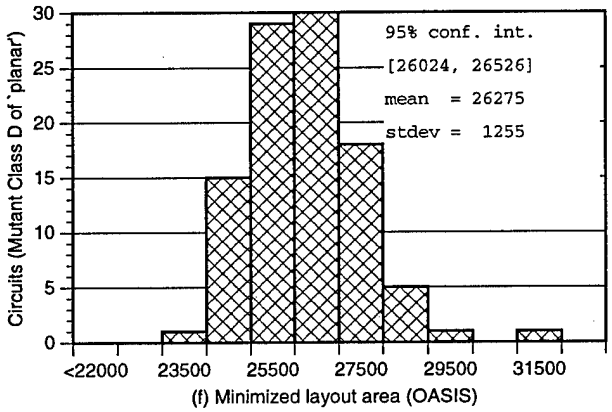

(c) Example of a net-to-cell channel optimized placement



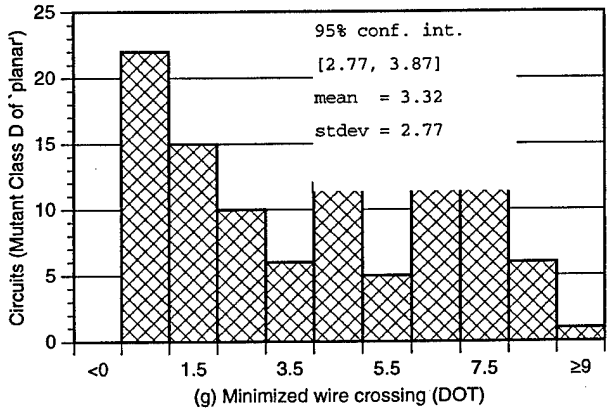
(d) Layout with OASIS (area=29568)

| | Initial | Final | Ideal |
|---|---|---|---|
| TWL | [51.54, 53.18] | [36.12, 37.02] | |
| | 52.36, 4.11 | 36.57, 2.26 | 28 |
| CWL | [9.49, 9.99] | [6.38, 6.62] | |
| | 9.74, 1.24 | 6.50, 0.61 | 5 |
| **TWC** | **[36.94, 39.72]** | **[2.77, 3.87]** | |
| | **38.33, 6.98** | **3.32, 2.77** | **0** |
| CWC | [12.03, 12.71] | [1.84, 2.54] | |
| | 12.37, 1.72 | 2.19, 1.74 | 0 |
| AWD | [2.23, 2.35] | [1.14, 1.20] | |
| | 2.29, 0.31 | 1.17, 0.18 | 0.33 |
| Width | [9.69, 10.04] | [6.40, 6.60] | |
| | 9.87, 0.92 | 6.50, 0.53 | 6 |
| Height | [4.30, 4.30] | [4.30, 4.30] | |
| | 4.30, 0.00 | 4.30, 0.00 | 4.3 |
| Area | [41.64, 43.20] | [27.50, 28.40] | |
| | 42.42, 3.94 | 27.95, 2.27 | 25.8 |

(e) Minimization for TWC




(f) Minimized layout area (OASIS)




(g) Minimized wire crossing (DOT)

Fig. 1. Subproblems in circuit layout: experiments with mutant Class 'D' of circuit 'planar'.

crossing of edges between net nodes and cell nodes. If an edge between two adjacent nodes crosses no cell separators (shown as dotted lines), we define its length as 0.5, and increase its length by 1 whenever it crosses a cell separator. The critical wire density is evaluated as the maximum number of edges crossing the cell separator.

(c) Wire crossing minimization of net-to-cell channels in (b) reduces total wire crossing to 0, wire length to 8.5, and the critical wire density to 1.

(d) An instance of the standard cell layout with OASIS [13] shows a number of wire crossings. In fact, upon translating the specific OASIS placement into the bipartite graph form on the same grid, we have 1 wire crossing in channels (0–1), 3 wire crossings in channels (1–2), and 1 wire crossing in channels (2–3). Inspecting the actual layout, we find more wire crossings in each of the chan-

nels.

(e) Table summarizing an experiment with 100 instances of the netlist of the 11-node planar circuit, each in random order, before submission to wire crossing minimization with DOT [12]. This is the W_ClassD circuit class, a special class of isomorphic mutants in the context of the design of experiments as introduced in [5]. We measure the initial and final value of sample mean, sample variance, and 95% confidence interval of the mean for several parameters we have already defined or will define later in the paper: total wire length (TWL), critical wire length (CWL), total wire crossing (TWC), width of placement (Width), height of placement (Height), and area of placement (Area).

(f) Histogram of circuit area for 100 instances of the netlist of the 11-node planar circuit, each in random order, submitted for layout optimization with OASIS [13].

(g) Histogram of circuit minimized wire crossing for 100 instances of the netlist of the 11-node planar circuit, each in random order, submitted for wire crossing minimization with DOT [12].

Conclusions that we draw from the observations recorded in Figure 1 include:

(a) The circuit is embedded onto a number of unit-size *grid regions*. A formal description is given in Section IV. The bipartite subgraph consisting of net nodes on the left and cell nodes on the right is called the *net-to-cell channel*. Similarly, the bipartite subgraph consisting of cell nodes on the left and net nodes on the right is called the *cell-to-net channel*.

(b) Random placement of nodes, either in a net-to-cell or a cell-to-net channel, can result in a large number of wire crossings.

(c) Wire crossing minimization, either in a net-to-cell or a cell-to-net channel, can reduce not only the wire length but also the wire density.

(d) Traditional layout optimization algorithms are not minimizing the cost of wire crossing.

(e) Wire crossing minimization of a complete circuit, not only a single channel segment, can improve a number of layout parameters in the bipartite graph model, not only wire crossing.

(f) Minimized circuit area for 100 instances of the netlist of the 11-node planar circuit, each listed in different random order, is a random variable and may be far from best possible.

(g) Minimized wire crossing for 100 instances of the netlist of the 11-node planar circuit, each listed in different random order, is a random variable. Only relatively few netlists have been returned with the known minimum wire crossing of 0. Unlike in the case of layout in (f), this distribution is *not normal* since more solutions are clustered to the left of the minimum value – lending support to our conjecture that optimizing the placement by minimizing wire crossing with a good algorithm may be as effective, if not more, than minimizing the wire length estimates.

Before we proceed with formal notation and definition, we digress with one more experiment, this time scaling the problem for a 2-layer graph minimization algorithm.

## III. Correlations in 2-Layer Graphs

The 2-layer subgraphs in the example shown in Figure 1 are too small to analyze correlations between grid-based variables such as wire crossing, wire length, and wire density.

The net-to-channel example in Figure 1(c) is a special case of a two-layer graphs from several families of parameterized 2-layer directed *sparse* graphs: Vn+1E3-n (E3-graphs) [14]. Properties of E3-graphs are: number of nodes at level 0 = n+1, number of 1-input nodes at level 1 = 3, number of 2-input nodes at level 1 = n, number of edges = 2*n + 3, number of wire crossings = 0.

We have chosen to analyze a graph with 65 net nodes, 131 edges, and 3+64 cell nodes (3 single-input nodes, 64 2-input nodes). We label its reference graph as V65E3-64. Next, we created 8 wiring signature-invariant equivalence classes, 100 circuits in each class (each with 65 net nodes, 131 edges, and 3+64 cell nodes) [5].

Here, we again analyze placements of the graph-isomorphic class W_ClassD. Evaluations of all 100 such random placements are shown in Figure 2(a-c): note the near-normal distribution of wire crossings (ranging from 3400 to 4800) and the near perfect correlation with wire length. As to the maximum wire density, it can vary from 59 to 89 wires.

Next, we submit the same 100 circuits to the drawing program DOT [12]. While much improved, only 14 out of 100 circuits could be placed with wire crossing of 0, optimum wire length of 194.5, and the maximum wire density of 1. The remaining circuits, while much better placed than the initial placement, are suboptimal and can still vary over a relatively large range.

In conclusion, the example in this section re-enforces our conjectures based on the earlier experiment in Figure 1:

- It is a *fallacy to rely on a single measurement of any benchmark circuit* – variations for many of the 'improvements' published to date may well be attributed to chance rather than any intrinsic improvement of the algorithm.
- The correlation coefficient between the total wire crossing and total wire length is high before and after wire crossing minimization; minimizing total wire crossing minimizes total wire length.
- The correlation coefficient between the total wire crossing and maximum wire density is much higher for random placement than it is for the near-optimal placement. It should be possible to 'improve' maximum wire density without a major change in wire crossing (and wire length).
- Any improvements in (1) the total wire crossing of all edges, and (2) the maximum wire crossing of all cell channel separator segments may translate to improved layout area and performance.

## IV. Notation and Definitions

The traditional graph-based models of a directed netlist are not effective for the problem we want to consider in this paper. On the other hand, a model of a netlist as a directed hypergraph is not unique. The important items in this work are the notion of cell level, levels of net pins, and *netspan*[2]. The *canonical form of a bipartite directed graph*, a multi-level graph structure of alternating sets of net nodes and cell nodes, is a simple transformation of the underlying netlist: levels of some of its pins are redefined, and a new type of cell node, a *feedthrough cell* is introduced [5].

Figure 3 relates to the definitions of a *grid region* $R(i,j)$. The vertical center-line of each region is called a *mid-line segment* and is either *free* or *occupied*. A mid-line segment can either be occupied by a single logic node, a single logic node and any number of feedthrough nodes, or any number of feedthrough nodes. The region is bounded on the left and the right by vertical lines called *net separator segments*. We

[2]For each net, netspan = $p_{max} - p_{min}$, where the two numbers denote the maximum and the minimum pin level of the net

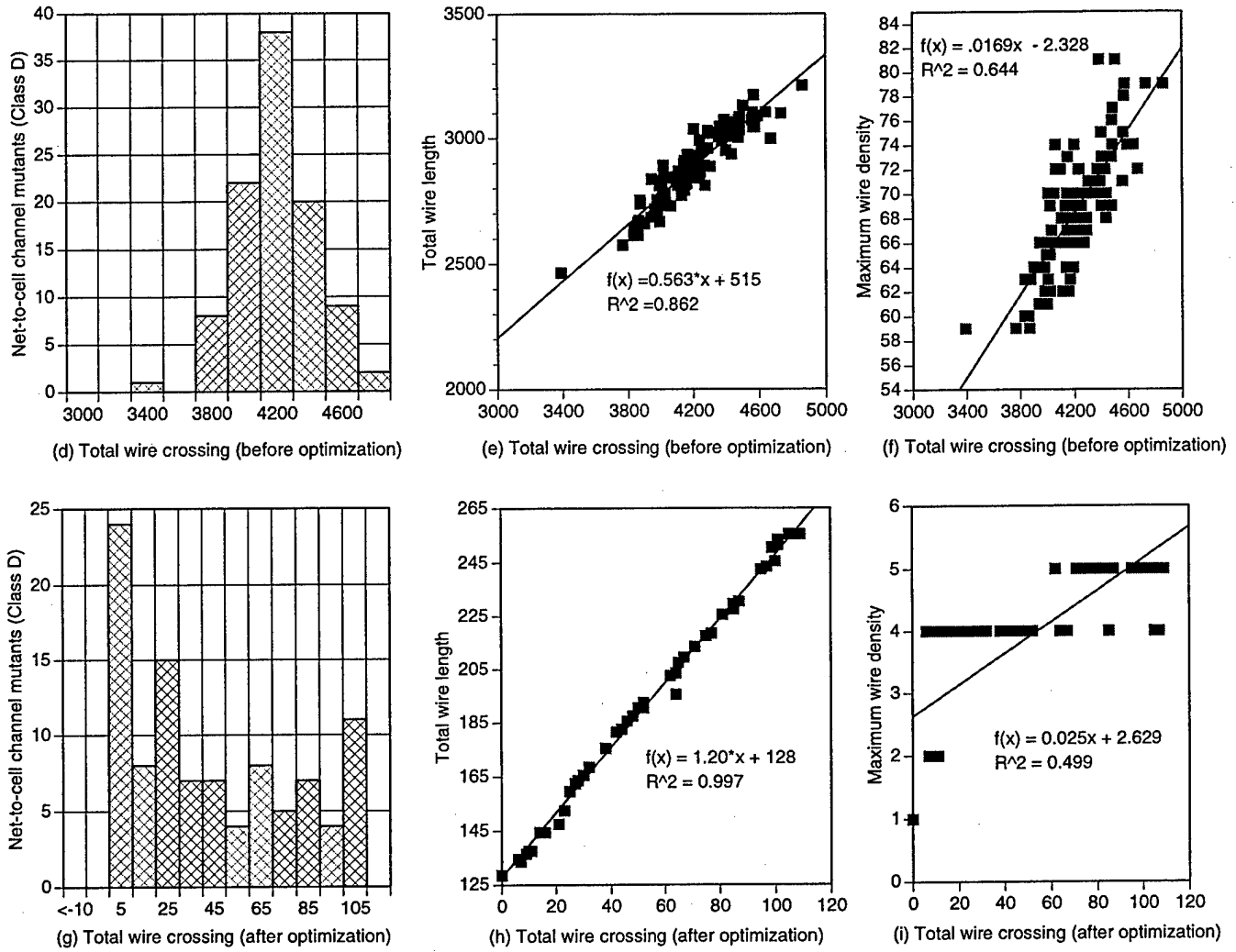



Fig. 2. Evaluation of random placements of V65E3-64.

index net nodes of the region either with net separator segment $i-1$ or net separator segment $i$, each being of height 1. The region $R(i,j)$ is bounded on the top and the bottom by *cell separator segments*, $j-1$ and $j$. We call the left half of a cell separator a *net-to-cell separator segment*, the right half a *cell-to-net separator segment*. Each has a width of 0.5. A *cell separator*, addressed with $j \in [0, j_{max}]$, is a union of non-overlapping cell separator segments. A *net separator*, addressed with $i \in [0, i_{max}]$, is a union of non-overlapping net separator segments. We may also refer to the index $i$ as the level of net nodes. A *mid-line* is a union of non-overlapping mid-line segments. Each mid-line segment is addressed by the index of the adjacent net separator on its right and the adjacent cell separator below.

**Elements of grid region $R(i,j)$.**

1. Two cell separator segments, shown in Figure 3 by dotted lines. One is at the boundary of grid regions $R(i,j-1)$ and $R(i,j)$ and the other at the boundary of grid regions $R(i,j)$ and $R(i,j+1)$.

2. An array of net nodes of type B. The $k^{th}$ element of this array is labeled $(i-1,j,k)_B$.

3. An array of cell nodes and feedthrough nodes of type A. The $k^{th}$ element of this array is labeled $(i,j,k)_A$.

4. An array of net nodes of type B, the $k^{th}$ element of which is labeled $(i,j,k)_B$.

5. Edges connecting the nodes in the region. The edges are classified as type A, if they connect nodes labeled $(i,j_1,k_1)_A$ and $(i,j_2,k_2)_B$. They are classified as type B,if they connect nodes labeled $(i-1,j_1,k_1)_B$ and $(i,j_2,k_2)_A$.

**Properties of grid region $R(i,j)$.**

**P1** In each grid region $R(i,j)$, there is at most one cell node.

**P2** For a cell in $R(i,j)$, there could be several permissible orientations. Each orientation defines the order of the pins of the cell.

**P3** It is possible to derive consistent co-ordinates for each pin of a node using its label and orientation. For an edge $e=(u,v)$, we define $y(u)$ and $y(v)$ as the co-ordinates of the pins it connects.

**P4** The cell separators in a channel induce pseudo–edges. We can use this property to calculate wiring density. These pseudo–edges connect nodes which are at the intersections of net separators and midlines. The co–

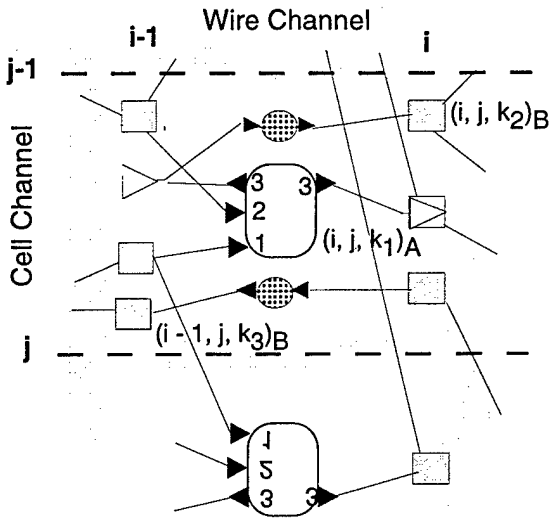



Fig. 3. Elements of grid region $R(i,j)$.

ordinates for these nodes are derived consistently with those of other nodes.

**P5** The length of each edge $e$, which connects nodes $(i_1,j_1,k_1)_A$ and $(i_2,j_2,k_2)_B$ is defined and evaluated as:

$$len(e) = \begin{cases} 0.5 & \text{if } j_1 = j_2 \\ 0.5 + |j_1 - j_2| & \text{otherwise} \end{cases} \quad (1)$$

**P6** Let $e_1 = (u_1, v_1)$ and $e_2 = (u_2, v_2)$ be edges such that $u_1$, $v_1$ are in the same wire channel and are pins of nodes with the same subscript ($A$ or $B$) as $u_2$ and $v_2$ respectively. We define a function $cross(e_1, e_2)$ as follows:

$$cross(e_1,e_2) = \begin{cases} 1 & \text{if } (y(u_1) - y(u_2))(y(v_1) - y(v_2)) < 0 \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

The wire crossing number for each edge $e$, is defined as follows:

$$wc(e) = \sum_{e_1 \in T(e)} cross(e, e_1) \quad (3)$$

where $T(e)$ is the set of all edges of the same type as $e$ in the same region as $e$.

**Figures of merit.** Let $E$ be the set of edges.

*Total wire length* is the sum of the wire length contribution of each edge.

$$wl\_tot = \sum_{e \in E} len(e) \quad (4)$$

*Critical wire length* is the length of the longest path starting at any primary input and ending at a primary output. This includes feedback primary inputs and outputs based on edges that break the cycle.

$$wl\_cri = Max_{(u_1 \in I, u_N \in O)}\{\sum_{i=1}^{N-1} len(u_i, u_{i+1})\} \quad (5)$$

where $I$ is the set of primary inputs, $O$ is the set of primary outputs and $(u_i, u_{i+1}) \in E$.

*Total wire crossings* is the total number of crossings in the placement.

$$wc\_tot = \frac{1}{2}\sum_{e \in E} wc(e) \quad (6)$$

The $\frac{1}{2}$ factor is due to the fact that each crossing is counted exactly twice, once for each of the two edges that produce it.

*Critical wire crossings* is the maximum number of crossings that appear on any path from an input to an output.

$$wc\_cri = Max_{(u_1 \in I, u_N \in O)}\{\sum_{i=1}^{N-1} wc(u_i, u_{i+1})\} \quad (7)$$

*Wiring Density* for a channel $C$, and a cell separator $s$, is the number of edges that cross $s$ in channel $C$.

$$wd(C,s) = \sum_{e \in E(C)} cross(s,e) \quad (8)$$

where $E(C)$ is the set of edges in channel $C$.

*Critical wire density* for a channel $C$ (cell-to-net or net-to-cell) is the maximum number of edges that cross any cell separator in that channel.

$$wd\_cri(C) = Max_{s \in S}(wd(C,s)) \quad (9)$$

*Total Wire Density* for a placement P is the sum of the critical wire densities for all the channels.

$$wd\_tot(P) = \sum_{C \in \mathcal{C}} wd\_cri(C) \quad (10)$$

where $\mathcal{C}$ is the set of all the channels in the placement.

*Average Wire Density* for a placement P, is the mean of the critical wire densities for all the channels.

$$awd(P) = \frac{wd\_tot(P)}{|\mathcal{C}|} \quad (11)$$

*Width* of a channel $C$ is defined as

$$width(C) = 1 + 0.5(wd\_cri(C)) \quad (12)$$

*Width* of the placement P is the sum of the widths of all the channels.

$$p\_width(P) = \sum_{C \in \mathcal{C}} width(C) \quad (13)$$

*Height* of a midline $M$ is defined as

$$height(M) = c(M) + 0.1f(M) \quad (14)$$

where $c(M)$ is the number of cell nodes in the midline, and $f(M)$ is the number of feedthrough nodes in the midline. The 0.1 factor is an empirical one.

*Height* of a placement P is the maximum height of any midline.

$$p\_height(P) = Max_{M \in \mathcal{M}}\{height(M)\} \quad (15)$$

where $\mathcal{M}$ is the set of all midlines in the placement.

*Area* of a placement P is the product of its height and width.

$$Area(P) = p\_height(P) \times p\_width(P) \quad (16)$$

**A Placed Netlist Format.** To describe the placement of a netlist in very simple terms, we found it convenient to extend,

```
.model FA2
.inputs a b c
.outputs r s
# wire cut 0 b d a
# cell cut 1 f d e
# wire cut 1 d f e h g c
# cell cut 2 r g h c.lev2
# wire cut 2 r g p h q c.lev2
# cell cut 3 p s q
# wire cut 3 s
.names h d r # given order
11 0
# wire channel h@1 d@1 r@2
# cell channel h@2 d@1 r@1
...
.names f e g # reverse order
11 0
# wire channel e@1 f@1 g@2
# cell channel e@2 f@1 g@2
...
.names c c.lev2 # given order
1 1
# wire channel c@1 c.lev2@2
# cell channel c@3 c.lev2@3
...
.end
```

Fig. 4. Illustrating annotated blif format for Place-Eval.

with structured comment lines, the `blif` format [15]. The example in Figure 4 highlights a section of a `placed blif` file that corresponds to the compacted and optimized placement of FA2, full adder circuit.

The additional comment lines `# wire cut i...` denote the order of net nodes in the net separator $i$. Similarly the comment lines `# cell cut i...` denote the order of cell nodes and feedthrough nodes in the midline $i$. With the description of any cell node a comment of `# given order` or `# reverse order` is inserted to indicate the orientation of the cell. Following the functional description of each node, there are 2 comment lines, one beginning with `# wire channel` which gives the wire channel that a cell node and its inputs occupy, the other begins with `# cell channel` which contains cell channel information about the cell node and its inputs. Feedthrough nodes like `c.lev2` are put in as buffers.

For example, the line `# wire channel e@1 f@1 g@2` signifies that the cell node `g` is located in the wire channel 2 and that there are two net nodes at its inputs, `e` and `f`, which are in the wire channel 1. The line `# cell channel e@2 f@1 g@2` signifies that the cell node `g` is located in the cell channel 2 and that there are two net nodes at its inputs, `e` and `f`, which are in the cell channels 2 and 1, respectively.

## V. TOCO Placement Algorithm

The acronym `TOCO` is based on the four steps of the proposed placement algorithm: *T*opological sort that places net nodes and cell nodes on the unit-size grid in level order, *O*ptimization of level order placement for minimum total wire crossing subject to the level order, *C*ompaction and folding of the level-order placement into a near-perfect square layout, *O*ptimization of compacted placement for minimum total wire crossing subject to the level order after compaction, *and* minimum total wire density, subject to order for minimum total wire crossing.

To minimize the total wire crossing, we currently use `DOT`, which implements a variant of the median heuristic [12]. To minimize the total wire density after total wire crossing minimization, we must assign net nodes and feedthrough cell nodes to grid regions such that critical wire density is minimized in *each* net-to-cell and cell-to-net channel – subject to node order imposed by the total wire crossing minimization algorithm. Our current approach consists of making a $k$-way partitioning assignment of $p$ net nodes and $q$ feedthrough nodes to $k$ grid regions in each channel, such that the wire crossing on the $k$-1 cell channel separator segments is minimized, subject to node order from total wire crossing minimization. The implementation of a total wire density optimization algorithm is in progress.

**Summary Overview of `TOCO`.**

**T** opological sort. This procedure may include the search for a minimal FVS in the circuit is sequential. It is the basis for generating the *initial placement* as a multi-level directed bipartite graph. This procedure assigns a level to every cell and net node in graph $G$.

**O** ptimization of placement $P(G)$ for minimization of wire crossings. At this point, the level of a node is still constrained by its level in the topologically sorted graph.

**C** compact and fold. Using the optimized placement obtained from the wire–crossing minimization algorithm, we *fold* the placement into a *compacted placement.* Basic rules are:

- *R1* Remove *all* feedthroughs. In our model, feedthroughs have no function other than that of routing nets which span more than one level.
- *R2* Count the cell nodes. If $N$ is the number of cell nodes, then the number of wire channels in the compacted form is $\lfloor\sqrt{N}\rfloor$. The number of cells in a wire channel, *wire channel capacity*, is determined similarly.
- *R3* If the current number of cell nodes in wire channel $i$ exceeds its wire channel capacity, then *fold forward* (move excess cell nodes at the top to the top of wire channel $i+1$.)
- *R4* If the current number of cell nodes in wire channel $i$ is less than its wire channel capacity, then *fold backwards* (move deficient cell nodes from the bottom of the next available wire channel to the bottom of wire channel $i$.)
- *R5* Insert feedthroughs for nets that span more than 1-wire channel.
- *R6* Save as a compacted placement, $P\_Compact(G)$.

**O** ptimization of placement of compacted $P(G)$. This consists of two distinct phases:

- *P1* Minimize the total wire crossing in the bipartite form embedding of the graph. Net nodes and cell nodes are constrained to the levels assigned to them by the compaction step.
- *P2* Minimize the total wire density in the bipartite form embedding of the graph. Net nodes and cell nodes are constrained to levels and the order assigned to them by the preceding wire crossing minimization phase. Wire crossing remains unchanged in this phase, only the total wire density is reduced.

**`TOCO` implementation and Pseudocode.** Implementations of various phases of the `TOCO` procedure have been optimized. Our annotated blif format seamlessly introduces net nodes for each cell node, as is required in the bipartite form. By using the same names for cell nodes and the net nodes they drive, the complexity of any phase in the procedure is reduced to that of searching in an array. Here our naming convention allows us to lexicographically sort arrays to make multiple simultaneous searches. The complexity of the procedure is dominated by these sorting calls and this can be done effi-

ciently in $O(nlogn)$.

Figure 5 shows the pseudo-code for the TOCO placement algorithm.

```
procedure TOCO_place(netlist)
   P(G) = Topological_Sort(netlist)
   P_opt(G) = Optimize(P(G))
   P_Compact(G) = Compact(P_Opt(G))
   Optimize(P_Compact(G))
end

procedure Compact(netlist)
   Remove_Feedthroughs(netlist)
   N = number_of_cell_nodes
   number_of_wire_channels = ⌊√N⌋
   foreach wire channel
     if cells(wire_channel) < capacity(wire_channel)
       Fold_Backward
     else
       Fold_Forward
   Insert_Feedthroughs
end
```

Fig. 5. Pseudo code of the TOCO placement algorithm.

## VI. Design of Experiments

The capability to synthesize a large number of WSI circuit mutants, based on wire perturbation classes, motivates us to examine the *sampling methods* that arise in the *design of experiments*. Such methods, first formalized in [16], have been adopted widely in many fields of science. In this paper, we adapt them to analyze the performance of important graph-based algorithms in the context of EDA. For each reference circuit, we propose to synthesize equivalence subclasses of circuit mutants, based on 0 to 100% perturbation. Each subclass contains 100 randomly chosen mutant circuits, each listed in a different random order. This sample size is large enough for the sampling distributions to be considered normal or nearly normal; the population parameters may be estimated closely by their corresponding sample statistics. The eight equivalence subclasses, labeled from $A$ to $H$, are defined in terms of the perturbations we use to generate each class. In order to encourage unbiased experiments with these classes, we have *permuted the perturbations* relative to the label assignments:

$$\{A, B, C, D, E, F, G, H\} = \text{permutation}\{0w, 1w, 2w, 5\%, 10\%, 20\%, 40\%, 100\%\} \quad (17)$$

In (17), WSI classes $A$-$H$ are defined either in terms of $q$-% wire perturbations or 0-wire, 1-wire, 2-wire ($0w, 1w, 2w$) perturbations. Here, we only disclose that the 0–wire perturbation class is `class_D`. We plan to identify the labels $A$-$H$ in terms of the respective perturbation classes in (17) later, once there are additional experiments reported by others and participants have the opportunity to meet and present their results at a joint session of a conference. More details about such plans can be found under http://www.cbl.ncsu.edu/experiments/.

A case study tutorial of average-case performance of two algorithms and their differences has demonstrated that up to six distinct equivalence classes of data are useful to render an unbiased comparison of two well-known sorting algorithms [17]. The long-term goal of this series of experiments, presently starting with eight equivalence classes, is to facilitate generation of similar comparisons for the more complex and diverse algorithms in EDA. Whatever may be decided about the most

```
(a) algorithms-vs-mutants-vs-classes

                      M-H_1  ....  M-H_k  ....  M-H_b
                      ------------------------------
                    ..............................
            M-B_1  ....  M-B_k  ....  M-B_b
            ---------------------------------
          M-A_1  ....  M-A_k  ....  M-A_b
          ---------------------------------
Alg_1-I     A_11-I ....  A_1k-I ....  A_1b-I
Alg_1-F     A_11-F ....  A_1k-F ....  A_1b-F
.....
Alg_j-I     A_j1-I ....  A_jk-I ....  A_jb-I
Alg_j-F     A_j1-I ....  A_jk-F ....  A_jb-F
.....
Alg_a-I     A_a1-I ....  A_ak-I ....  A_ab-I
Alg_a-F     A_a1-F ....  A_ak-F ....  A_ab-F

(b) classes-vs-mutants (for a given algorithm Alg_j)

Alg_j-F    M-._1  ....  M-._k  ....  M-._b
          ----------------------------------
M-A          A_j1-F ....  A_jk-F ....  A_jb-F  {M-A_j-F}
M-B          B_j1-F ....  B_jk-F ....  B_jb-F  {M-A_j-F}
....
M-H          H_j1-F ....  H_jk-F ....  H_jb-F  {M-H_j-F}
          ----------------------------------
         {M-._j1-F} ..{M-._jk-F} ..{M-._jb-F}

(c) statistics of algorithms-vs-mutant classes

           M-A        M-B     .....  M-H
          ---------------------------------
Alg_1-I   {M-A_1-I}  {M-B_1-I} ... {M-H_1-I}
Alg_1-F   {M-A_1-F}  {M-B_1-F} ... {M-H_1-F}
.....
Alg_j-I   {M-A_j-I}  {M-B_j-I} ... {M-H_j-I}
Alg_j-F   {M-A_j-F}  {M-B_j-F} ... {M-H_j-F}
.....
Alg_a-I   {M-A_a-I}  {M-B_a-I} ... {M-H_a-I}
Alg_a-F   {M-A_a-F}  {M-B_a-F} ... {M-H_a-F}
```

Fig. 6. Data structures and classes for the proposed experiments.

suitable number of equivalence classes through wider participation later on, the class of 0-wire perturbations will remain important. As demonstrated in this paper as well as earlier [18], the objective functions used in a number of graph-based algorithms can be very sensitive to the *order of nodes* in the graph, even when graphs are isomorphic. In our experiments, the 100 netlists in the 0-wire perturbation class are simply isomorphic instances of the reference netlist in a randomized order.

Paraphrasing the context of the traditional *treatments and blocks* [19], we propose to archive data in the context of *algorithms and equivalence class mutants* as shown in Figure 6(a). For each of the '$a$' algorithms we consider '$b$' mutants in one of the equivalence classes in (17). For each algorithm `Alg_j` and mutant `M-X_k`, $X \in \{A, \ldots, H\}$, we record two observations: the *initial value* of the objective function tuple `X_jk-I`, and the *final value* of the objective function tuple `X_jk-F`. The initial value corresponds to a *placebo treatment* of the mutant `M-X_k`: it is the value of the objective function *before* engaging the algorithm to optimize it. The final value corresponds to the optimized value of the objective function *after* engaging the algorithm to optimize it.

A number of analyses can be performed once data is archived as shown in Figure 6(a) and only a few are discussed in this paper. For the most part, we shall concentrate

on analyzing data as presented in Figure 6(b). In particular, for samples associated with each algorithm Alg_j and mutant class M-X, $X \in \{A, \ldots, H\}$, we evaluate the 95% confidence interval of the sample mean, and the sample variances as tuples {M-X_j-I} and {M-X_j-F} respectively. We summarize such evaluations in the form shown in Figure 6(c). The next section provides representative summaries of data samples we generated and archived under http://www.cbl.ncsu.edu/experiments/.

## VII. Experiments

This section summarizes experiments based on four equivalence classes of circuit mutants of the reference circuit C1355 [20, 21]. The four classes, with 100 mutant circuits in each class, are based on wiring signature-invariant perturbations as defined in Section VI: W_Class_B, W_Class_C, W_Class_D, and W_Class_E. Notably, W_Class_D is the 0-wire perturbation class – all netlists in this class are isomorphic.

**Context of experiments.** The context in which we propose to perform placement experiments is shown in Figure 7. The flow may be executed as follows:

1. prepare netlists of appropriate equivalence classes;
2. place any number of equivalence class netlists with any placer e.g TOCO;
3. archive results of placements in a common *grid-based format*, such as suggested in Figure 4;
4. process grid-based placements by either or both:
   (a) a common placement evaluator (e.g. place_eval that can process results of any grid-based placement and produce a *standardized report* in terms of grid-based layout parameters, optionally mapped to technology-specific parameters from the library database;
   (b) a common router that can process results of any grid-based placement and produce a layout and a *standardized report* in terms of technology-specific layout parameters.

Ideally, one may be able to correlate the grid-based parameter reported by place_eval, such as total and critical wire crossings, total and critical wire length, total and critical wire density, to comparable parameters measured in actual layouts. If the correlations are reasonable, the grid-based model may be calibrated for fast estimation of cell-based layouts.

Finally, execution of placement experiments, with different algorithms, as suggested in Figure 7, will allow us to study their behavior and improve them further.

**Figure 8 – Histograms.** Here we contrast layout parameters reported by the placement and routing tool OASIS [13] and grid-based parameters optimized by TOCO and reported by place_eval – for 100 instances of the mutant circuits in W_Class_B. The coefficient of variation (mean/stdev) for layout area reported by OASIS is 4.4%, and 3.8% for the minimized wire crossing reported by TOCO. It would be useful to know whether the placements from TOCO would improve the layout beyond the one currently shown for OASIS.

It is clear that that wirelength and total layout area, as reported by OASIS, are closely correlated. Minimizing the wirelength does minimize the area. The square of the correlation coefficient of total area to total wirelength as reported by OASIS is 0.835. However, current results with TOCO show that the correlation of total wire length and total wire crossing is not high. We attribute this behavior to the sub–optimal assignment of net nodes to the respective cell channels. We expect the correlation to improve once we complete the implementation of the channel wire density minimization algorithm.

**Figure 8 – Table.** Here we summarize results of experiments with 4 classes of circuit mutants of C1355. Each entry reports the 95% confidence interval for the mean, the sample mean, and the sample variance. The initial values refer to values of parameters after compaction with TOCO. The final values are after the wire crossing optimization phase. For all these classes, the procedure dramatically reduces the total wire length (TWL), critical wire length (CWL), total wire crossing (TWC) and critical wire crossing (CWC). However the improvements in other parameters such as average wire density (AWD) and width are not as dramatic. We expect to report improved results for the average wire density (AWD) and width in the final version of this paper; upon completion of the channel wire density minimization algorithm implementation.

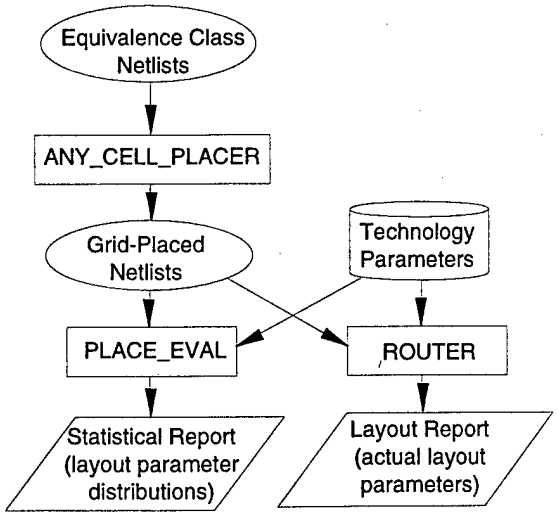
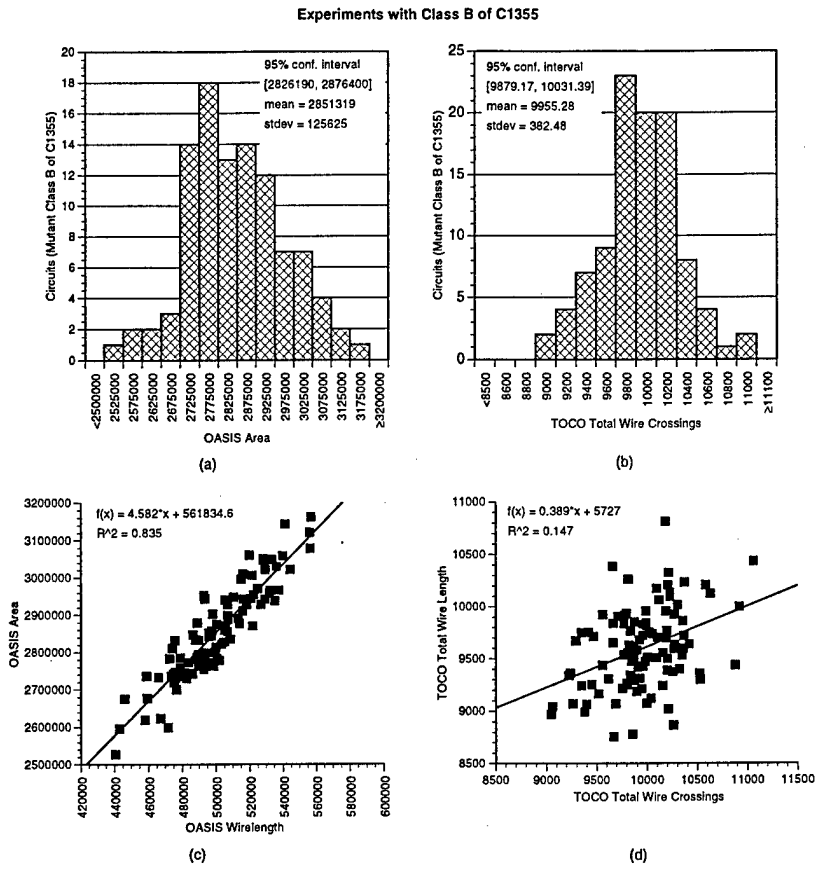



Fig. 7. Proposed context of placement experiments.

**Statistical interpretations.** The reader should observe the sensitivity of algorithms when evaluating the 0-wire perturbation class (W_Class_D) in the table of Figure 8 and elsewhere [5]. Experiments implemented in this paper were designed to begin addressing issues such as

- Consider, for a given mutant class, (1) sample mean and standard deviation of the *unoptimized* objective function, and (2) sample mean and standard deviation of the objective function *optimized* via algorithm Alg_j. We have to decide: (H0) is the difference in the means due to chance, or (H1) is it due to the effect of algorithm Alg_j?
- Consider, for a given algorithm Alg_j, (1) sample mean and standard deviation of the *optimized* objective function in terms of mutant class $A$, and (2) sample mean and standard deviation of the *optimized* objective function in terms of mutant class $B$. We have to decide: (H0) is the difference in the means due to chance, or (H1) is it due to differences of the two mutant classes?
- Consider, for a given mutant class (1) sample mean and standard deviation of the objective function *optimized* by algorithm Alg_j1, and (2) sample mean and standard deviation of the objective function *optimized* by algorithm Alg_j2. We have to decide: (H0) is the difference in the means due to chance, or (H1) is it due to different performances of the algorithms Alg_j1 and Alg_j2?

To encourage wider participation, complete tables of *all* data samples summarized in this paper have been archived on our web site (http://www.cbl.ncsu.edu/experiments/). The archives are updated periodically with *additional experiments* and cases of more detailed statistical analyses [14]. The web site provides an open forum to interested researchers for further sampling, *tests of significance and hy-*




| | W_Class_B | W_Class_C | W_Class_D | W_Class_E |
|---|---|---|---|---|
| TWL–I | [50004.8, 50668.2] | [48548.4, 49173.2] | [44596.5, 45163.9] | [47207.6, 47814.1] |
| | 50336.53, 1658.37 | 48860.77, 1562.05 | 44880.19, 1418.66 | 47510.85, 1516.32 |
| TWL–F | [ 9522.90, 9677.24] | [ 9419.45, 9587.73] | [ 8140.59, 8322.09] | [ 9123.57, 9295.27] |
| | 9600.07, 387.81 | 9503.59, 422.84 | 8231.34, 456.05 | 9209.42, 431.41 |
| CWL–I | [ 1018.47, 1044.35] | [ 1024.21, 1046.71] | [ 1048.33, 1070.91] | [ 1038.85, 1062.91] |
| | 1031.41, 65.04 | 1035.46, 56.52 | 1059.62, 56.73 | 1050.88, 60.46 |
| CWL–F | [ 266.49, 273.91] | [ 268.43, 275.53] | [ 246.02, 253.14] | [ 269.95, 277.23] |
| | 270.20, 18.67 | 271.98, 17.82 | 249.58, 17.87 | 273.59, 18.30 |
| TWC–I | [ 37094.69, 37427.63] | [ 36291.43, 36637.33] | [ 33880.97, 34182.31] | [ 35471.07, 35811.87] |
| | 37261.16, 836.56 | 36464.38, 869.08 | 34031.64, 757.11 | 35641.47, 856.28 |
| TWC–F | [ 9879.17, 10031.39] | [ 9029.02, 9184.40] | [ 2934.66, 3083.30] | [ 7326.78, 7457.10] |
| | 9955.28, 382.48 | 9106.71, 390.42 | 3008.98, 373.45 | 7391.94, 327.45 |
| CWC–I | [ 1510.96, 1544.08] | [ 1538.95, 1569.53] | [ 1516.07, 1537.59] | [ 1553.45, 1582.39] |
| | 1527.52, 83.20 | 1554.24, 76.82 | 1526.83, 54.08 | 1567.92, 72.71 |
| CWC–F | [ 611.70, 629.54] | [ 585.29, 604.89] | [ 279.20, 293.36] | [ 517.79, 533.43] |
| | 620.62, 44.82 | 595.09, 49.22 | 286.28, 35.59 | 525.61, 39.28 |
| AWD–I | [ 13.28, 13.42] | [ 13.30, 13.44] | [ 13.84, 13.94] | [ 13.50, 13.64] |
| | 13.35, 0.36 | 13.37, 0.33 | 13.89, 0.27 | 13.57, 0.34 |
| AWD–F | [ 13.03, 13.29] | [ 13.09, 13.33] | [ 11.72, 12.02] | [ 12.84, 13.14] |
| | 13.16, 0.65 | 13.21, 0.62 | 11.87, 0.73 | 12.99, 0.73 |
| Width–I | [ 314.09, 317.27] | [ 314.61, 317.49] | [ 326.36, 328.74] | [ 318.99, 321.93] |
| | 315.68, 7.97 | 316.05, 7.22 | 327.55, 5.98 | 320.46, 7.39 |
| Width–F | [ 308.68, 314.33] | [ 309.90, 315.30] | [ 279.92, 286.30] | [ 304.61, 310.99] |
| | 311.50, 14.23 | 312.60, 13.59 | 283.11, 16.04 | 307.80, 16.04 |
| Height–I | [ 35.62, 35.84] | [ 35.64, 35.84] | [ 35.27, 35.47] | [ 35.67, 35.89] |
| | 35.73, 0.56 | 35.74, 0.51 | 35.37, 0.50 | 35.78, 0.57 |
| Height–F | [ 35.62, 35.84] | [ 35.64, 35.84] | [ 35.27, 35.47] | [ 35.67, 35.89] |
| | 35.73, 0.56 | 35.74, 0.51 | 35.37, 0.50 | 35.78, 0.57 |
| Area–I | [ 11210.71, 11350.01] | [ 11231.66, 11362.14] | [ 11528.71, 11639.77] | [ 11400.81, 11533.15] |
| | 11280.36, 350.00 | 11296.90, 327.85 | 11584.24, 279.04 | 11466.98, 332.52 |
| Area–F | [ 11017.38, 11248.24] | [ 11064.37, 11285.37] | [ 9893.25, 10133.79] | [ 10888.50, 11143.50] |
| | 11132.81, 580.04 | 11174.87, 555.28 | 10013.52, 604.36 | 11016.00, 640.70 |

Fig. 8. Summary of placement results in TOCO for mutant classes of circuit C1355.

*potheses*, and *statistical inference* of existing data and benchmarks, as well as for contributing new cases of benchmarks, new data from experiments, and new cases of statistical analysis. The site will maintain contributions of partici-

pants either as hyperlinks to data and documents on participant's web site, or new archives will be created under http://www.cbl.ncsu.edu/experiments/.

## VIII. Conclusions

The main conjecture in this paper, that *good column-based placements can be produced by minimizing two wire crossing numbers*, requires access to an open routing tool that will accept a user-defined placement. Ideally, more than a single routing tool should be used in such a study. We are looking for a partner to assist us in this process. The large number of mutant circuits in equivalence classes provides a sound basis for the design of benchmarking experiments that can prove or disprove this conjecture at a sound level of significance.

Collaborative web-based experiments are being initiated under http://www.cbl.ncsu.edu/experiments/. The proposed placement evaluator `place_eval` will process results of any grid-based placement and produce a *standardized report* in terms of grid-based layout parameters. We hope that the approach, after testing by several users and some calibration, will prove useful.

Acknowledgments. We appreciate the access to tools used in this research: DOT from AT&T Research.

## References

[1] W. Sun and C. Sechen. Efficient and Effective Placement for Very Large Circuits. *IEEE Transactions on Computer–Aided Design of Integrated Circuits and Systems*, 14(3):349–359, March 1995.

[2] C. Sechen. Average Interconnection Length Estimation for Random and Optimized Placements. In *IEEE Intl. Conf. Computer-Aided Design*, pages 190–193, November 1987.

[3] T. N. Bui and S. Lee. On the Mincut Bipartite Arrangement Problem. In *IEEE Intl. Conf. Computer-Aided Design*, pages 466–469, November 1987.

[4] F. T. Leighton. New Lower Bound Techniques for VLSI. *Math. Systems Theory*, 17:47–70, 1984.

[5] D. Ghosh, N. Kapur, J. Harlow, and F. Brglez. Synthesis of Wiring Signature-Invariant Equivalence Class Circuit Mutants and Applications to Benchmarking. In *Design Automation & Test - Europe (DATE'98)*, February 1998.

[6] J. N. Warfield. Crossing Theory and Hierarchy Mapping. *IEEE Trans. Sys., Man, and Cybernetics*, pages 505–523, July 1977.

[7] M. R. Garey and D. S. Johnson. Crossing Number is NP-complete. *SIAM J. Algebraic Discrete Methods*, 4:312–316, 1983.

[8] K. Sugiyama and S. Tagawa and M. Toda. Methods for Visual Understanding of Hierarchical System Structures. *IEEE Transactions on Systems, Man, and Cybernetics*, 11(2):109–125, 1981.

[9] P. Eades and N. C. Wormald. Edge Crossings in Drawings of Bipartite Graphs. *Algorithmica*, 11:379–403, 1994.

[10] T. Catarci. The Assignment Heuristic for Crossing Reduction. *IEEE Transactions on Systems, Man, and Cybernetics*, 25(3):515–521, March 1995.

[11] M. Jünger and P. Mutzel. 2–Layer Straightline Crossing Minimization: Performance of Exact and Heuristic Algorithms. *Journal of Graph Algorithms and Applications (JGAA)*, 1(1):1–25, 1997. Available at `http://www.mpi-sb.mpg.de/~mutzel/mpireports/`.

[12] E.R. Gasner *et el.* A Technique for Drawing Directed Graphs. *IEEE Trans. Software Engg.*, 19:214–230, 1993. Software available at `http://www.research.att.com/sw/tools/graphviz/`.

[13] K. Kozminski, (Ed.). OASIS2.0 User's Guide. MCNC, Research Triangle Park, N.C. 27709, 1992.

[14] F. Brglez (Ed.). Collaborative Archives of On-Going Experiments with Algorithms in EDA: Taxonomy of Data Samples and Analysis of Multiple Comparisons. Technical report, 1997. Also available at `http://www.cbl.ncsu.edu/publications/`.

[15] SIS – Release 1.1. UC Berkeley Software Dist., Sept. 1992.

[16] R. A. Fisher. *Statistical Methods, Experimental Design, and Scientific Inference.* Oxford University Press, 1993. Reprinted, with corrections, from earlier versions, 1925-1973.

[17] F. Brglez. A Tutorial on Design of Experiments to Benchmark Algorithms in EDA. Technical report, 1997. Also available at `http://www.cbl.ncsu.edu/publications/`.

[18] M. R. Hartoog. Analysis of Placement Procedures for VLSI Standard Cell Layout. In *Design Automation Conference*, pages 314–319, July 1986.

[19] K. A. Brownlee. *Statistical Theory and Methodology In Science and Engineering.* Krieger Publishing, 1984. Reprinted, with revisons, from second edition, 1965.

[20] F. Brglez and H. Fujiwara. Special Session on ATPG (Also introducing 'A Neutral Netlist of 10 Combinational Benchmark Circuits'). In *Int. Symp. On Circuits and Systems*, 1985. A basis for a benchmark directory ISCAS85, now archived at `http://www.cbl.ncsu.edu/benchmarks/`.

[21] S. Yang. Logic Synthesis and Optimization Benchmarks User Guide. Technical report, MCNC, RTP, NC, Jan. 1991. Report and benchmarks archived at `http://www.cbl.ncsu.edu/benchmarks/`.